Embedded Evolutionary Robotics: The (1+1)-Restart-Online Adaptation Algorithm

Jean-Marc Montanier, Nicolas Bredeche

Chapter 11
Embedded Evolutionary Robotics: The (1+1)-Restart-Online Adaptation Algorithm

Jean-Marc Montanier and Nicolas Bredeche

Abstract. This paper deals with online onboard behavior optimization for a autonomous mobile robot in the scope of the European FP7 Symbrion Project. The work presented here extends the (1+1)-online algorithm introduced in [4]. The (1+1)-online algorithm has a limitation regarding the ability to perform global search whenever a local optimum is reached. Our new implementation of the algorithm, termed (1+1)-restart-online algorithm, addresses this issue and has been successfully experimented using a Cortex M3 microcontroller connected to a realistic robot simulator as well as within an autonomous robot based on an Atmel ATmega128 microcontroller. Results from the experiments show that the new algorithm is able to escape local optima and to perform behavior optimization in a complete autonomous fashion. As a consequence, it is able to converge faster and provides a richer set of relevant controllers compared to the previous implementation.

11.1 Introduction

Let's imagine an autonomous mobile robot tailored for exploration. This robot could be dropped in a wide variety of unknown environments, from a dense tropical forest to an exhausted gold mine abandoned 100 years ago. Even before starting to explore its environment, this kind of robot would need to be able to adapt to its immediate surrounding (i.e. figuring out what shape and/or what behavior is most fitted to sustain its energy level). In this set-up, the robot control architecture is first driven by the specific, unpredictable, properties of the environment.

This paper focuses on the design of a control architecture for an autonomous mobile robot in an unknown environment. For this type of set-up, design approaches can range from hand-crafted reactive behavior [3] to optimal control approaches [9].

Jean-Marc Montanier · Nicolas Bredeche
TAO - Univ. Paris-Sud, INRIA, CNRS
LRI, Bat. 490, F-91405 Orsay, France
e-mail: forname.name@lri.fr

S. Doncieux et al. (Eds.): New Horizons in Evolutionary Robotics, SCI 341, pp. 155–169.
springerlink.com

The chosen approach depend on the problem to be solved. In the aforementionned situation, it is difficult, if not impossible, to a priori specify the environment and the task at hand. This implies that most of the existing approaches are not fitted. This is a typical problem in Robotics that may be addressed with learning and optimization[10, 21]. Moreover, we address the problem where little is known about the objective function. This means that the task is poorly described as a single efficiency measure (e.g. minimize energy consumption, maximize exploration, etc.), which is often delayed and noisy.

In this scope, Evolutionary Robotics provides optimization algorithms based on Evolutionary Algorithms. Evolutionary Robotics [9, 16] (”ER”) takes inspiration from nature by combining two partly antagonist mechanisms. On the one hand, *selection* of the most fitted individuals tends to ensure convergence of the algorithm toward the most fitted solution. On the other hand, *variation* over the properties of selected individuals through mutation and recombination tends to provide new original solutions. The general framework of these algorithms, termed Evolutionary Algorithms (”EA”), has been applied to a wide variety of problems [6]. In Evolutionary Robotics, EA is used as an optimizer for Artificial Neural Network architectures. Artificial Neural Network are used to control autonomous robots in a wide variety of task, from navigation (non-linear task) to swarm coordination (cooperation task)[1].

In ER, in order to compute the quality (or *fitness*) of a given genotype, the corresponding phenotype is created (e.g. an artificial neural network for robot control with optimized weights). This phenotype is *evaluated* in the environment to assess the performance of the resulting robot behavior. This evaluation methodology, is used on a *population* of *genotypes*. The (usually) better genotypes are selected and go through a variation process so as to renew the population. This process is then iterated until a termination criterion is matched (e.g. maximum number of evaluation, performance, etc.). This approach is usually referred as *off-line ER*.

While off-line ER can be used to address non-linear control problems (or poorly defined objective function), it fails to provide a continuous autonomous optimization process. This is because control over the initial conditions for genome evaluation is required. Therefore, either costly human interventation or the use of simulation [13] is needed. Moreover, evaluation of a genome requires reliability, ie. the fact that one evaluation session must be relevant with regards to the problem at hand.

Embodied ER, introduced in [8], is a sub-field of ER that precisely addresses the problem of changing environments without constant human manutention. In this setup, the Evolutionary Algorithm runs within the robot (or group of robots), acting as an embedded optimization algorithm. Embedded is defined as both online (the adaptation/learning process never stops) and onboard (the optimization algorithm and evaluation process are part of the control loop). To date, only few, but promising, works have adressed this topic: [7, 11, 12, 14, 15, 17, 20, 23, 24, 25, 26]. Running an embedded EA within a single robot provides strong advantages regarding

[1] See [10] for an introduction.

continuous adaptation and autonomy with regards to a human supervisor. However, this also emphasizes some specific issues:

- Unknown fitness landscape: the typical fitness landscape in ER is both multi-modal (many local minima) and partly neutral (many close genotype perform in a similar way). One reliable assumption (named Strong Causality) is that small variations in the genotypic space implies small variations in the fitness value [18]. A direct consequence is that any reliable algorithm should be able to perform both local search (to exploit this strong causality property) and global search (to avoid the pitfall of multi-modality);
- Evaluation reliability: the environmental condition vary over time depending on the robot location. Therefore, performance assessment (ie. fitness) of one genome might be completely different from one starting location to another (e.g. starting in a narrow bridge or starting in the middle of an empty arena). This is the problem of noisy fitness evaluation. A great number of independant evaluation are required to assess for the ”true” performance of one genome;

The (1+1)-online adaptation algorithm described in [4] has been shown to address these issues. Its ability to perform continuous adaptation efficiently has been demonstrated in the same paper[2] The (1+1)-online algorithm is described as a genetic algorithm based on the (1+1)ES [19]. This algorithm uses only two genomes: a champion and a challenger. Some specific properties are employed so as to address online adaptation:

- **Local and global search :** A mutation operator is used to produce a child from a parent. This mutation operator is able to do both local and global search. A gaussian distribution $N(0,\sigma)$ is used. The switching between local and global search is done by modifying the value of σ. If this value is low, few modifications will be done to the genome, and the search will remain local. If the value of σ is high, the genome will be strongly modified, and the research will go global. On one hand, the value of σ is set to a low value when a new champion is found. This ensure a local search around a known good to solution in order to improve it. On the other hand, the value of σ is increased when the challenger is assessed worst than the current champion. This second mechanism ensure that the search will go more global if the current champion is in a local optima.
- **Re-evaluation :** Individuals may get lucky or unlucky during evaluation depending on the envrionment at hand. This is a typical problem related to fitness noise. An efficient solution is to reevaluate individuals, as proposed by Beyer [2].The reevaluated fitness overwrites the fitness of the champion. This is done to promote individuals with a low variance in their performances. One of the drawback of the overwriting method is that good individuals could be replaced by inferior but lucky individuals. If an individual is lucky during its first evaluation but has a low mean fitness it will not survive next-reevaluations. As a consequence, the evolutionary algorithm won’t be stuck with bad individuals.

[2] The demonstration was done on a single e-puck robot in Player/Stage, running a Cortex M3 micro-controller.

- **Recovery :** This work assumes the evolutionary algorithm should run without human intervention. It implies no repositioning of the robot after each evaluation of one individual. For example, a genome may be evaluated starting from completely different initial conditions, such as in front of a wall or in a tight corner. To avoid penalization of good genomes, a *recovery period* is introduced. During this time, the robot behavior is not considered for evaluation (ie. "free of charge"), which favors genomes that display good performance whatever the starting position.

In this paper, we present an analysis of the global search feature of this algorithm. From this analysis, we identify a problem that negatively impacts the search. The basic idea is that the previous implementation of the (1+1)-online algorithm restrains, possibly drastically, the search space considered. This imply a limitation in the efficiency of the global search mechanism. This problem is described and a new algorithm, termed (1+1)-restart-online is devised. Preliminary experiments in simulation are described and show that the new algorithm actually performs a larger global search. Therefore the new algorithm, avoid the pitfall of getting stuck in a local optima for a long time. Moreover, this paper describes the implementation and successful evaluation of this new algorithm on a real robotic hardware set-up, a four wheels Bioloid mobile robot.

11.2 Extending the (1+1)-Online EA

This section shed some light on an intrinsic limitation of the (1+1)-online algorithm. Under very specific conditions (multi-modal objective function with few or no amount of noise), the adaptation process is slowed down. Then, an extension of the previous algorithm is described. This extension retains the properties of the original algorithm, and addresses the problem identified.

11.2.1 Limits of (1+1)-Online

In [4], the efficiency of the (1+1)-online algorithm has been shown. One of its main properties is to rely on a tunable gaussian mutation operator σ to switch between local and global search (see section 11.1). The update scheme of σ seems to be relevant in most cases, but it has a major drawback : only regions with better performing genomes can be considered. Figure 11.1 illustrates the shrinking effect of the search region considered. On this figure the fitness values of all genomes is shown (for the sake of simplicity, we assume this is a minimization task for a one dimension only problem). In this example, the current champion may be replaced *only* by a challenger which is *under* the dashed line. This holds for both local and global search. In this typical set-up, this isn't a relevant strategy as the probability to randomly jump to the relevant region is very low. In comparison, it is more appealing to pick a genome from which local search may slowly but surely, lead to the best genome.

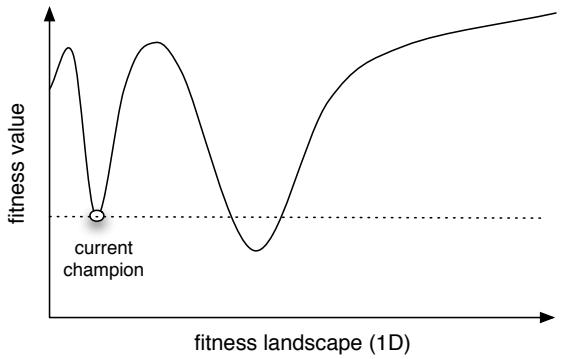

Fig. 11.1 Deceiving fitness landscape (minimization task).

The modification of σ is a good candidate to find new individuals. When it is increasing the search goes more global. But at some point the search area is too constrained. It becomes more interesting to simply restart the whole optimization process in order to obtain an unconstrained global search. To some extent, this problem may not occur in all situations. Firstly, this problem would never occur when optimizing a convex objective funtion, which is unfortunately quite scarce in this set-up. Secondly, a very noisy objective function may cope with this problem. That is because any good performing individual may eventually be re-evaluated with a low fitness value. Therefore the whole search space will be considered all over again – this was indeed the case in the experiments shown in [4].

11.2.2 The (1+1)-Restart-Online Algorithm

Escape from local minimum is a classical problem for the global search algorithms, and has been studied in different fields. A popular method is the restart strategy as used in some state of the art Evolution Strategies algorithm [1]. With this approach, the algorithm is restarted, either with similar or different parameters, whenever the search is identified as stalled. This approach provides interesting results on multimodal problems as it ensures global convergence (ie. the algorithm tends to explore the whole search space through random search, as it is exactly what restarting is about).

In order to implement restart in the $(1+1) - restart - online$ algorithm, the restart criterion has to be considered. Options are mostly limited to the two following:

- **Monitoring the value of σ :** If σ reaches a maximal predefined value, a local minimum has been attained and the search is going global. To be sure that the algorithm will never be blocked in a local minimum, it can be restarted as soon as σ reaches its maximal value.

- **Limiting the number of champion reevaluations:** Whenever a champion isn't replaced, no better individuals have been found in its neighborhood. Thus, surviving many re-evaluations assesses the robustness of the champion with regards to both other challenger genomes and to the environment. Therefore, a high number of re-evaluations can be used to detect a good performing genome, but also that the search is stalled.

However, some issues remain to be addressed. On the one side, relying on the value of σ alone to restart the algorithm is too constraining: the maximum value for σ may be reached even if the champion was not reevaluated yet (ie. the champion may be unreliable). Moreover, even if the current champion has been successfully reevaluated while σ was increasing, it may still be improved by mutations. On the other side, if the champion survives many reevaluations, it is a good and reliable individual that will be hard to replace.

As a consequence, our implementation is to consider the number of reevaluations as a restart criterion in the (1+1)-restart-online algorithm described by algorithm 1. Hence, whenever restart is triggered, the algorithm is re-initialized, storing the current champion in a hall-of-fame archive, and setting a random genome as the new starting point. In the long term, this tends to converge towards a uniform sampling of the genotypic space.

11.3 Experiments and Results

This section presents the experimental set-up used to evaluate the performance of the (1+1)-restart-online algorithm. Results and preliminary experiments are also described and discussed.

11.3.1 Hardware Set-Up

The experimental evaluation has been conducted using a popular robotic microcontroller: a Cortex M3 board with 256 kb memory. The Cortex board runs a robot simulated by Symbricator3D, a physics-based 3D simulator developped within the Symbrion project and based on delta3d[3] (An Open Source game engine which can be used for physics-based simulations). After N time-steps, the evaluation of the current controller is complete. The controller parameters are replaced with values from a new genome. As described in the previous section, the new genome is evaluated from the location the previous controller left it in.

Figure 11.2 illustrates the experimental set-up with a Cortex board connected to the computer running the simulator based on delta3d. The simulated robot is equiped with two screws and 8 distance sensors (two per side). Details of the shape of the robot can be seen in figure 11.3. The maze environment and the cortex board are shown in figure 11.2.

[3] http://www.delta3d.org/

Algorithm 1. The (1+1)-RESTART-ONLINE evolutionary algorithm.

```
for evaluation = 0 to N do
  if random() < P_reevaluate then
    if reevaluation_count < reevaluation_max then
      Recover(Champion)
      Fitness_Champion = RunAndEvaluate(Champion)
      reevaluation_count = reevaluation_count + 1
    else
      σ = σ_min
      Champion = RandomGenome()
      Fitness_Champion = 0
      Challenger = RandomGenome()
      Fitness_Challenger = 0
      reevaluation_count = 0
    end if
  else
    Challenger = Champion + N(0, σ) {Gaussian mutation}
    Recover(Challenger)
    Fitness_Challenger = RunAndEvaluate(Challenger)
    if Fitness_Challenger > Fitness_Champion then
      Champion = Challenger
      Fitness_Champion = Fitness_Challenger
      σ = σ_min
    else
      σ = σ · 2
    end if
  end if
end for
```

The robot is controlled by a simple perceptron with 9 input neurons (8 IR distance sensor values and one bias neuron) and 2 output neurons (translational and rotational velocities, which are combined to give actual actuator values).

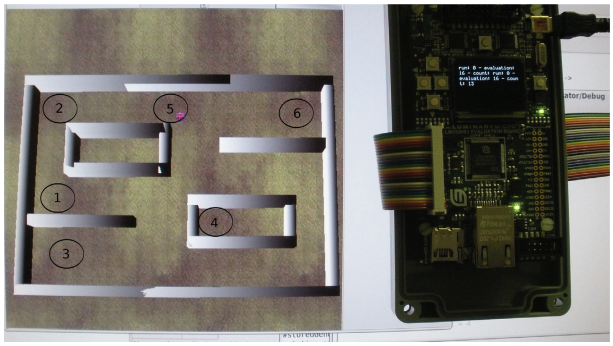

Fig. 11.2 The experimental set-up: the Cortex M3 board connected to Symbricator3d. The numbers show the reference starting positions for validating the Hall of Fame.

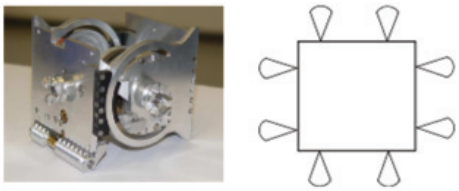

Fig. 11.3 Details of the Symbricator robot. (a) robot design (b) position of distance sensors (from above).

11.3.2 Experimental Set-Up

The objective function used is inspired from a well-known and classic fitness function first described in [16]:

$$fitness(x) = \sum_{t=0}^{n} V_t * (1 - V_r) * (1 - D_M)$$

where V_t is the translational speed, V_r is the rotational speed, and D_M the value of the most active distance sensor. All values are normalized between 0 and 1. More details about the parameters setting are given in appendix I. Distance sensors returns higher value when they are close to a wall. Therefore, individuals achieve high fitness value while moving fast and forward and avoiding walls.

The (1+1)-restart-online algorithm has been evaluated with a restart parameter fixed at 7 reevaluations. In order to compare the true performances of individuals obtained with (1+1)-online and (1+1)-restart-online, one Hall-of-Fame per experiment is computed from the results of the simulations. A Hall-of-Fame contains the best individuals (ie. the genome champions) from a given experiment. Champion genomes are ranked thanks to the sum of the re-evaluated fitness obtained during the experiments (ie. the larger the fitness value and the larger the number of re-evaluations, the better genome).

As the adaptation process could go on forever, the maximal number of evaluations is fixed to 600 for both experiments. Afterwards, the following experimental protocol is used to compare the best individuals from the Hall-of-Fames: every champions from the two Hall-of-Fames are evaluated from 6 predefined starting positions[4] shown in figure 11.2 and each evaluation lasts 120 time steps (ie. long enough to evaluate the quality of behaviors in a wide range of situations). The motivation of this validation protocol is to provide fair comparison between genomes.

[4] The starting position number 4 is an extreme case where the robot is tested in a hard environment never seen before.

11.3.3 Experimental Results

Figure 11.4 shows the course of evolution during a critical run of the (1+1)-online algorithm. Evaluations are denoted on the x-axis. The y-axis is divided in two parts. The top half shows the fitness of the current champion in green dashed line. The bottom half shows the number of re-evaluations of the current champion (downwards; the lower the line, the higher the number of re-evaluations). The red vertical markers near the x-axis indicate whenever a new champion is employed, i.e., when the challenger outperforms the current champion. During this run a good champion has been found at evaluation 180, and hasn't been replaced until evaluation 538 (after 64 reevaluation). Due to the use of another simulator this problem hasn't been detected in [4]. Differences between the two simulators are important with regards to noise between evaluations. This is a typical illustration of the problem identified in this paper with the original (1+1)-online algorithm: a less noisy set-up is prooved to be deceitful for the original algorithm.

Figure 11.5 shows evolution dynamics of a run of the (1+1)-restart-online algorithm. On this run the two main features of this algorithm – reevaluation and restart – are displayed. The reevaluation procedure is used at evaluation 132 on a lucky individual found at evaluation 126. After this reevaluation the champion obtain a low fitness and is soon replaced. In this run the restart procedure is used at evaluation 368, to replace a robust champion. According to preliminary experiments, the champion of evaluation 368 could still be improved. This imply that the restart strategy could be triggered later.

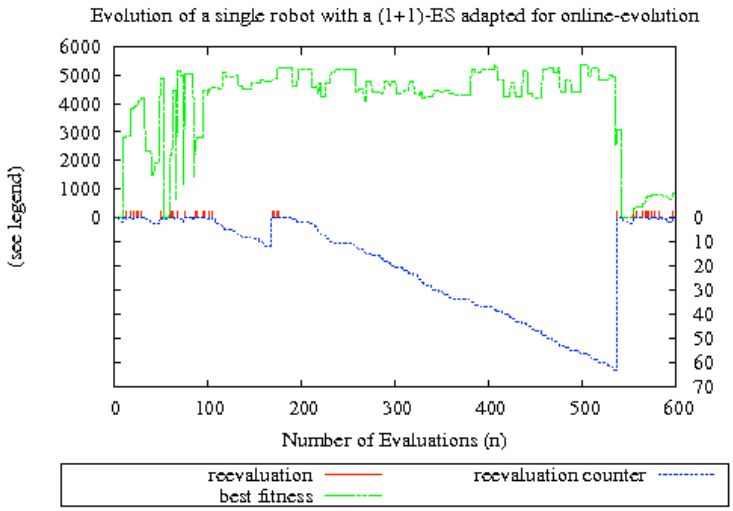

Fig. 11.4 Evolutionary dynamics of a critical run of the (1+1)-online algorithm.

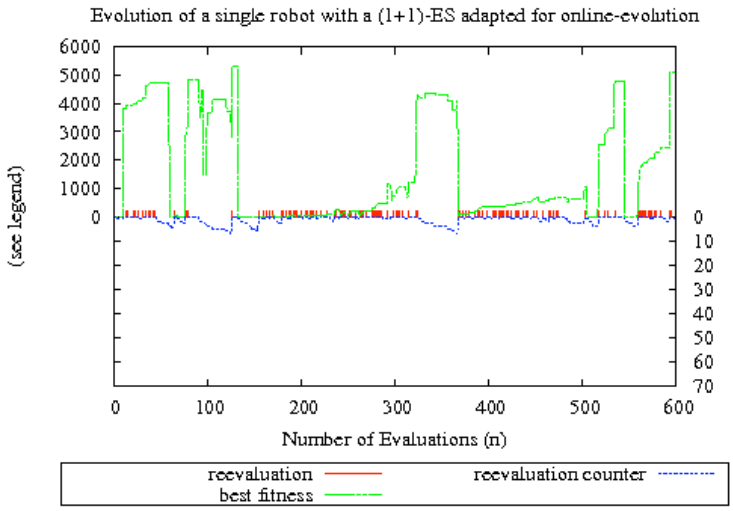

Fig. 11.5 Evolutionary dynamics of a run of the (1+1)-restart-online algorithm.

11.3.4 Hall-of-Fame Analysis

As described in section 11.3.2, two Hall-of-Fames were extracted from the results of the experiments. Each Hall-of-Fame is computed out of 14 independant runs of 600 evaluations. There are 1691 individuals in the Hall-of-Fame obtained by running the (1+1)-online algorithm. In comparison, 2158 individuals are in the Hall-of-Fame obtained by running the (1+1)-restart-online algorithm. This difference is a desired effect of the (1+1)-restart-online algorithm. The restart feature favors exploration by saving unnecessary reevaluations of champions whenever the algorithm is stalled. As a consequence, the $(1+1)-restart-online$ provides faster (in term of number of evaluation) the same number of Hall-of-Fame indivudals.

Performances of the best individuals generated by the (1+1)-restart-online algorithm and by the (1+1)-online algorithm are compared. As described in section 11.3.2, every individuals from the Hall-of-Fames are evaluated from six pre-defined positions (i.e. to provide comparable test cases). For each individual the mean performance obtained from those 6 positions has been computed. The figure 11.6 reports the distribution of individuals with respect to their fitness. The x-axis shows the different fitness obtained during the validation of the 628 best individuals of each Hall-of-Fame. The y-axis shows the number of individuals with the same fitness. It is clear that there is no loss of efficiency with the (1+1)-restart-online algorithm.

Therefore, the $(1+1)-restart-online$ algorithm is as reliable as the $(1+1)-online$ and faster - which is a key feature whenever ressources are limited.

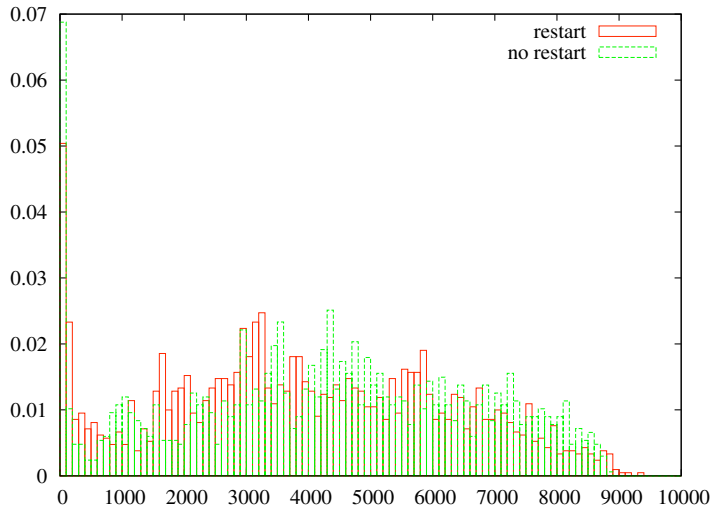

Fig. 11.6 Fitness density of the best individuals produced by the (1+1)-online algorithm, and the (1+1)-restart-online algorithm.

11.3.5 Real Robot Experiment

The (1+1)-restart-online has been tested on an autonomous four-wheels Bioloid robot. The Bioloid kit provides robotic parts and an ATmega128 microcontroller with 128Kb of memory[5]. Figure 11.7 shows the robot used in this work. It is equiped with 4 motors, and 7 distance sensors. The 7 red arrows in figure 11.7 shows the orientations of the distance sensors. The controller of the robot is a feedforward neural network with 8 inputs (7 distance sensors and 1 bias) and 2 outputs (left and right motor velocities). The two left wheel velocities are controled by the same output neuron, and the two right wheel velocities by the other output neuron. The fitness function used is the same as the one described in section 11.3.2. Each individual is recovering during 60 time step (7 seconds) and is evaluated during the 60 following time step (7 seconds). As in section 11.3.2 the restart threshold is fixed to 7 re-evaluations. The whole experiment lasted 1 hour and 10 minutes, which was more than sufficient to get examples of robust behaviors.

Figure 11.7 (b) shows the experimental set-up.

The algorithm provides similar results to what has been already shown in the previous experiment. The behavioral traces of the first two best evolved controllers from the Hall-of-Fame are illustrated in figures 11.8 and 11.9. These two control architectures efficiently avoid walls with simple yet efficient behaviors. The best controller (figure 11.8) is faster when moving in straight line, and displays sharper

[5] `http://www.atmel.com/dyn/products/product_card.asp?part_id=2018`

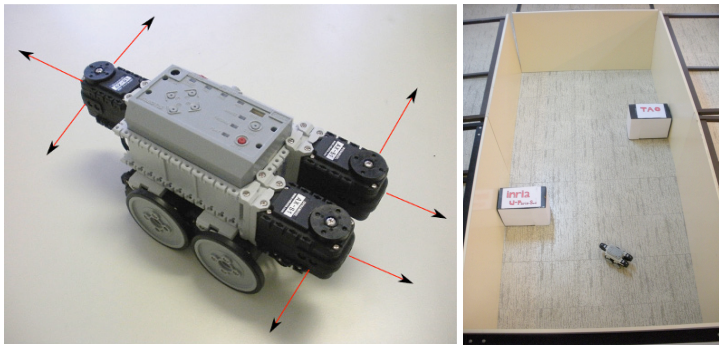

Fig. 11.7 (a) The robot and the directions of the 7 distance sensors, (b) the environment.

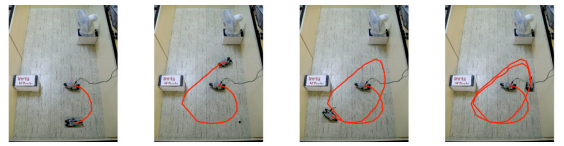

Fig. 11.8 Example of behavior for the best evolved controller.

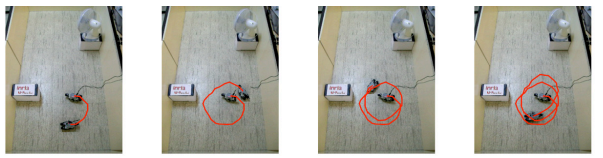

Fig. 11.9 Example of behavior for the 2nd best evolved controller.

turn trajectories. Other genomes have been empirically evaluated (not shown here) and display the same kind of behaviors as these two, with minor differences concerning the sharpness of the turn and/or the speed of the robot.

An important feature of our algorithm is also demonstrated here : the online nature of the algorithm makes it possible to easily avoid the reality gap [13]. Indeed, the algorithm needed exactly the same amount of work from the experimenter in simulation and reality. Moreover, *neither* human intervention *nor* external remote control was ever needed during the whole experiment with the real robot. Of course, this assumption must be taken with care as the fitness considered here is a rather simple one, chosen to focus on the validation of the algorithm features rather than on its ability to solve a complex problem.

11.4 Conclusion and Perspectives

In this paper, the problem of online onboard behavior adaptation for a single autonomous mobile robot has been adressed. Precisely, the (1+1)-online adaptation algorithm presented in [4] is studied. A limitation of this algorithm is identified and analysed regarding its ability to perform global search in the space of possible solutions. A new algorithm is described, termed (1+1)-restart-online. It is shown to efficiently address the trade-off between local and global search, by relying on a restart procedure whenever the algorithm is stuck in a local optimum. This restart procedure makes it possible to address a previous design flaw by relaxing some of the required constraints over the search space considered.

This algorithm has been evaluated both with real robotic hardware connected to a robot simulation as well as with a real robot. Results obtained have demonstrated that this new algorithm is actually able to provide a wider exploration of the search space, making it possible to visit many more local optima than previous implementation. Therefore, the probability to end up in a global optimum is increased, but also the diversity of obtained candidate solutions is increased. Moreover, this algorithm can be straight-forwardly used within a real robot in a complete autonomous fashion, providing a key-feature to relieve the expert from unnecessary and fastidious control over the experiment.

Perspectives from this work first concerns a careful study of the experimental parameters and have already been started in [5]. Also, an in-depth analysis of the distribution of the performance from all individuals in the Hall-of-Fame should be conducted. Moreover, the new restart feature in the algorithm must be carefully studied as there exists a possible trade-off in balancing the previous global search strategy and the new restart strategy. This trade-off can be reformulated as favoring global search over avoiding re-convergence towards already visited local optima. As a consequence, choosing between the two strategies clearly depends on both the shape of the fitness landscape and the actual local minimum.

Future works will also address the problem of noisy fitness evaluation by extending the $(1+1)$ stragegy into a variation of more complex strategies, from building challenger genomes out of family of successful genomes or distribution-based estimation of the relevant genotypic regions to explore (as in estimation of distribution algorithms). This roughly means that a reservoir, or a distribution, of champion genomes will be considered rather than only a single champion genome in order to build new candidate genome to be evaluated. Also, the extension towards multi-robots is an interesting perspective: one may consider the current adaptation algorithm to act within one island of a distributed evolutionary algorithm. In this set-up, each robot/island runs an embodied evolutionary adaptation algorithm. Additionally, the best genomes may be exchanged from one island to another, as in the well-known GA island model [22].

Acknowledgments

This work was made possible by the European Union FET Proactive Intiative: Pervasive Adaptation funding the Symbrion project under grant agreement 216342.

References

1. Auger, A., Hansen, N.: A restart cma evolution strategy with increasing population size. In: Proceedings of the IEEE Congress on Evolutionary Computation, CEC 2005 (2005)
2. Beyer, H.G.: Evolutionary algorithms in noisy environments: Theoretical issues and guidelines for practice. In: Computer Methods in Applied Mechanics and Engineering, pp. 239–267 (1998)
3. Braintenberg, V.: Vehicles: Experiments in Synthetic Psychology. MIT Press, Cambridge (1986)
4. Bredeche, N., Haasdijk, E., Eiben, A.: On-line, on-board evolution of robot controllers. In: Collet, P., Monmarché, N., Legrand, P., Schoenauer, M., Lutton, E. (eds.) EA 2009. LNCS, vol. 5975, pp. 110–121. Springer, Heidelberg (2010)
5. Eiben, A., Haasdijk, E., Bredeche, N.: Symbiotic Multi-Robot Organisms: Reliability, Adaptability, Evolution. In: Levi, P., Kernbach, S. (eds.) Symbiotic Multi-Robot Organisms. Cognitive Systems Monographs, vol. 7, pp. 361–382. Springer, Heidelberg (2010)
6. Eiben, A.E., Michalewicz, Z. (eds.): Evolutionary Computation. IOS Press, Amsterdam (1998)
7. Elfwing, S.: Embodied Evolution of Learning Ability. PhD thesis, KTH School of Computer Science and Communication, SE-100 44 Stockholm, Sweden (November 2007)
8. Ficici, S., Watson, R., Pollack, J.: Embodied evolution: A response to challenges in evolutionary robotics. In: Wyatt, J.L., Demiris, J. (eds.) EWLR 1999. LNCS (LNAI), vol. 1812, pp. 14–22. Springer, Heidelberg (2000)
9. Floreano, D., Husbands, P., Nolfi, S.: Evolutionary robotics. In: Siciliano, B., Khatib, O. (eds.) Handbook of Robotics, pp. 1423–1451. Springer, Heidelberg (2008)
10. Floreano, D., Mattiussi, C.: Bio-Inspired Artificial Intelligence: Theories, Methods, and Technologies. In: Intelligent Robotics and Autonomous Agents. MIT Press, Cambridge (2008)
11. Floreano, D., Schoeni, N., Caprari, G., Blynel, J.: Evolutionary Bits'n'Spikes. In: Standish, R.K., Beadau, M.A., Abbass, H.A. (eds.) 8th International Conference on the Simulation and Synthesis of Living Systems (Alife 8). MIT Press, Cambridge (2002)
12. Haroun Mahdavi, S., Bentley, P.J.: Innately adaptive robotics through embodied evolution. Auton. Robots 20(2), 149–163 (2006)
13. Jakobi, N., Husband, P., Harvey, I.: Noise and the reality gap: The use of simulation in evolutionary robotics. In: Morán, F., Merelo, J.J., Moreno, A., Chacon, P. (eds.) ECAL 1995. LNCS, vol. 929, Springer, Heidelberg (1995)
14. Koenig, L., Jebens, K., Kernbach, S., Levi, P.: Stability of on-line and on-board evolving of adaptive collective behavior. In: European Robotics Symposium 2008. Springer Tracts in Advanced Robotics, vol. 44, pp. 293–302. Springer, Heidelberg (2008)
15. Nehmzow, U.: Physically embedded genetic algorithm learning in multi-robot scenarios: The pega algorithm. In: Prince, C., Demiris, Y., Marom, Y., Kozima, H., Balkenius, C. (eds.) Proceedings of The Second International Workshop on Epigenetic Robotics: Modeling Cognitive Development in Robotic Systems, LUCS, Edinburgh, UK, vol. 94 (August 2002)

16. Nolfi, S., Floreano, D.: Evolutionary Robotics: The Biology, Intelligence, and Technology of Self-Organizing Machines. MIT Press/Bradford Books, Cambridge, MA (2000)
17. Perez, A.L.F., Bittencourt, G., Roisenberg, M.: Embodied evolution with a new genetic programming variation algorithm. In: ICAS, pp. 118–123 (2008)
18. Rechenberg, I.: Evolutionstrategie: Optimierung Technisher Systeme nach Prinzipien des Biologischen Evolution. Fromman-Hozlboog Verlag, Stuttgart (1973)
19. Schwefel, H.-P.: Numerical Optimisation of Computer Models. Wiley, New York (1981)
20. Simões, E.D.V., Dimond, K.R.: Embedding a distributed evolutionary system into population of autonomous mobile robots. In: Proceedings of the 2001 IEEE Systems, Man, and Cybernetics Conference (2001)
21. Thrun, S., Burgard, W., Fox, D.: Probabilistic Robotics. MIT Press, Cambridge (2005)
22. Tomassini, M.: Spatially structured evolutionary algorithms: Artificial evolution in space and time. In: Natural Computing Series (2005)
23. Usui, Y., Arita, T.: Situated and embodied evolution in collective evolutionary robotics. In: Proceedings of the 8th International Symposium on Artificial Life and Robotics, pp. 212–215 (2003)
24. Walker, J.H., Garrett, S.M., Wilson, M.S.: The balance between initial training and lifelong adaptation in evolving robot controllers. IEEE Transactions on Systems, Man, and Cybernetics, Part B 36(2), 423–432 (2006)
25. Watson, R.A., Ficici, S.G., Pollack, J.B.: Embodied evolution: Distributing an evolutionary algorithm in a population of robots. Robotics and Autonomous Systems 39(1), 1–18 (2002)
26. Wischmann, S., Stamm, K., Wörgötter, F.: Embodied evolution and learning: The neglected timing of maturation. In: Almeida e Costa, F., Rocha, L.M., Costa, E., Harvey, I., Coutinho, A. (eds.) ECAL 2007. LNCS (LNAI), vol. 4648, pp. 284–293. Springer, Heidelberg (2007)

Implementation Details of the (1+1)-Online Algorithm

- Each individual runned during 120 time step (60 time step of recovering and 60 time step of evaluation).
- 600 evaluations per experiments.
- 14 runs with (1+1)-online algorithm and 14 runs with restart (1+1)-online algorithm.
- random individual at a random position, at the beginning of a run.
- $P_{reevaluate}$ is set to 0.2.
- σ initial value is set to 1 and may range from 0.01 to 4.
- genes value in [-4,+4].